

The 6th International Summer Workshop on Multimodal Interfaces

eNTERFACE '10

amsterdam :: The Netherlands :: 12.07 - 06.08.2010 ::

hosted by ISLA-ISIS, University of Amsterdam

Project #7

Audio Visual Speech Recognition

Final Presentation

06.08.2010

The 6th International Summer Workshop on Multimodal Interfaces

eNTERFACE '10

amsterdam :: The Netherlands :: 12.07 - 06.08.2010 ::

hosted by ISLA-ISIS, University of Amsterdam

PROJECT MEMBERS

Hakan Erdoğan
Saygın Topkaya
Berkay Yılmaz
Umut Şen
Alexey Tarasov

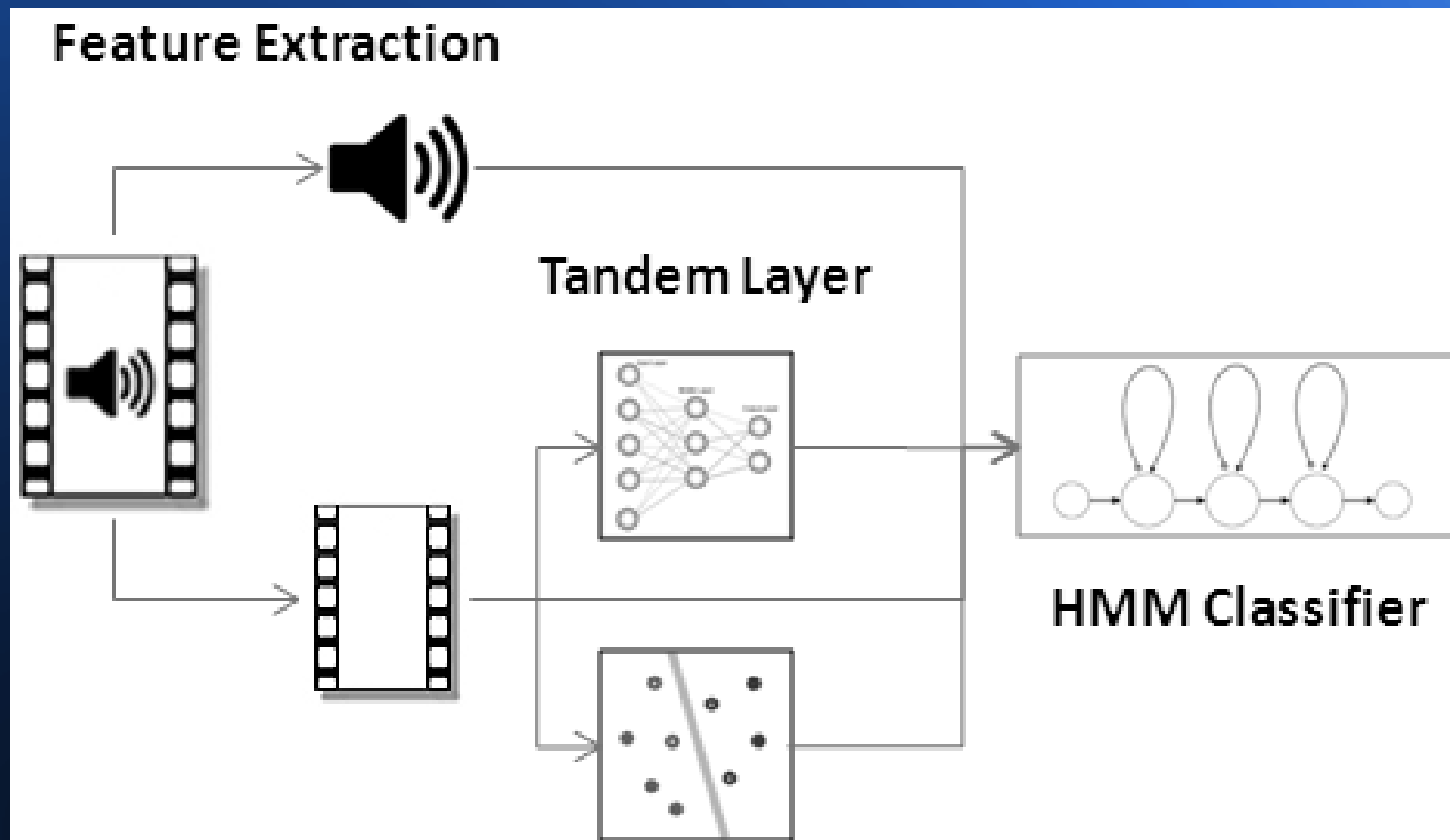
Presentation Outline

1. Project Information
2. Proposed System (Tandem MSHMM)
3. Project Aim & Steps
4. Overview of Implementation:
 - Data Acquisition
 - Feature Extraction
 - Decoding / Recognition
5. Progress & Problems / To Be Done...

Project Information

- Under noise speech only performance may vary; special steps taken to reduce effects of noise
- Audio data can be supported by visual data (i.e. lip/mouth information)
- Two modalities can be fused at feature level or decision level
- Observations can be supported with other first level classifier results

Proposed System



Project Aim

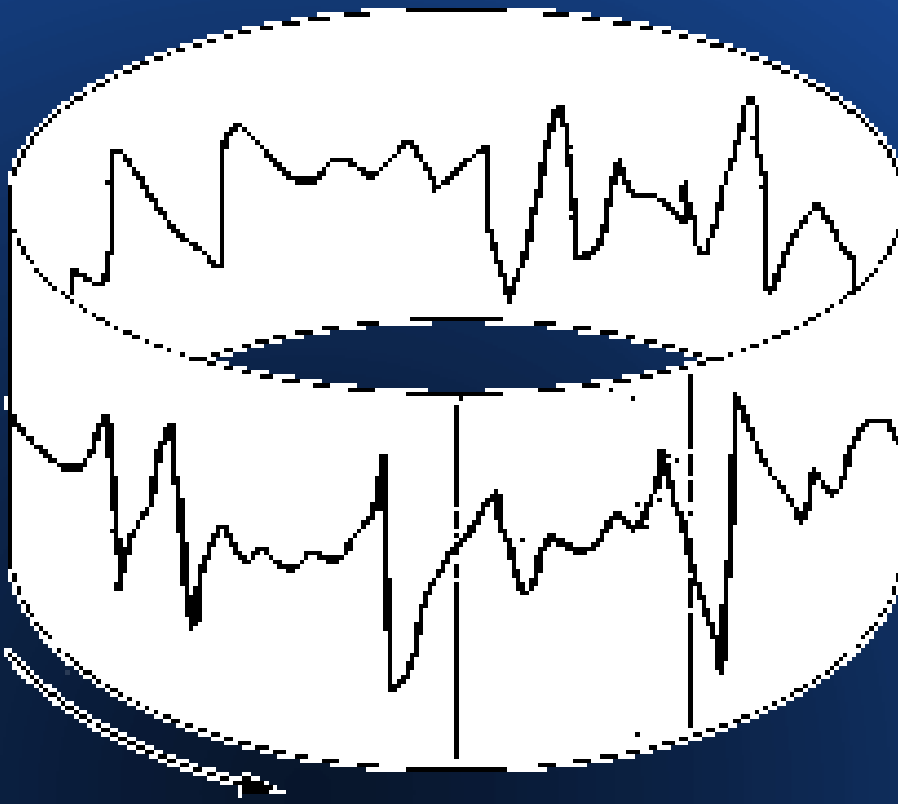
- Perform speech recognition in real-time
- Use both audio & video data
(e.g. input from a camera + microphone)
- Use Multi-Stream HMMs
- Give different weights to audio and video streams at different noise levels
- Extract & model video tandem features as additional streams of the MSHMM

Project Steps

- Data acquisition
- Feature extraction
for both audio and video
- First level classifiers
for video tandem streams
- MSHMM on streams of;
Audio + Video + Video Tandem(s)

Implementation

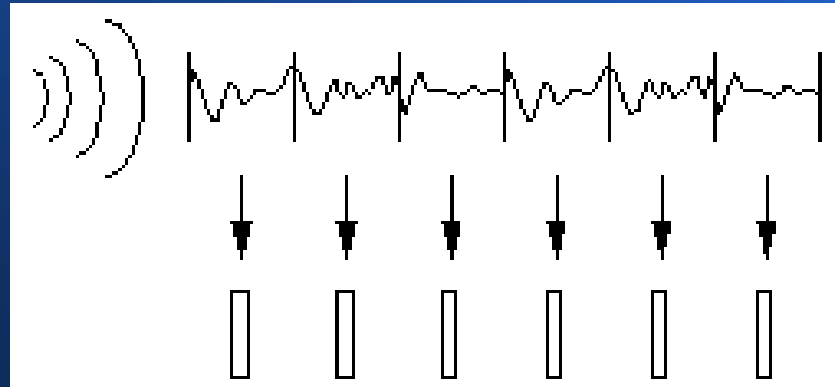
Model a circular buffer of frames as a complex class with:



- Audio/Video data
- Visual ROI
- Audio/Video features
- Tandem Features

Continuously HMM decoding on
buffer elements → frames

Audio Data Acquisition



- Acquire audio data with *PortAudio*
- Extract MFCC features with *libXtract*

Video Data & ROI (Nose Detection)

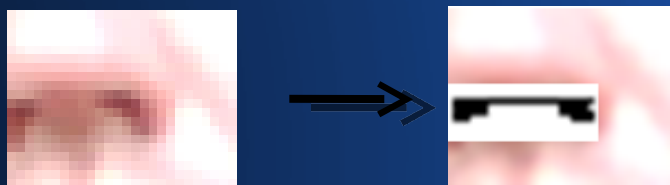
Use OpenCV to acquire data

- Original frame is converted into gray level
- Viola-Jones object detectors (haar cascades) are used to extract face and nose regions
- Nose region is searched in the middle part of the face



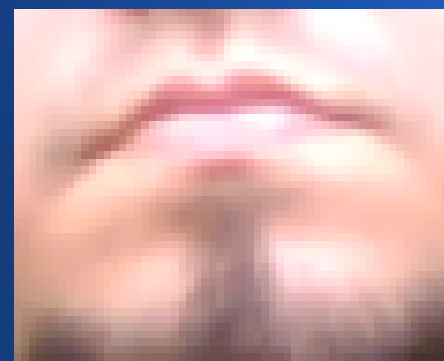
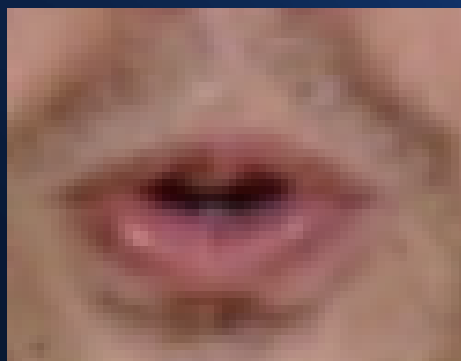
Video Data & ROI (Nostril Detection)

- Nostrils determine the upper bound of mouth region
- Nostrils are found in nose region by thresholding
 - Best threshold is found looking at nose region histogram
 - Both ends of the darkest nose region are accepted as approximate nostrils. Center of nose holes becomes the upper boundary of mouth region



Video Data & ROI (ROI Extraction)

- Left and right lip corners are determined from face region using a basic assumption
- With the knowledge of top, left and right ends of the lip region, bottom end is found by a square region assumption:



Video Data & ROI (Tracking The Lip Region)

- Once the ROI is extracted, it is tracked using the nose region
 - Nose region's appearance doesn't change a lot
 - Correlation of the nose with the face is found in every frame
 - The point giving the highest peak is compared with the previous highest peak
 - Difference of the peak locations gives an estimate of head motion
- So it is possible to track the mouth region without finding it in every frame
- Average translation of several (~ 10) previous frames is used to smooth the motion

Video Data & ROI (Resetting The ROI)

- During the tracking, the process may be reset and everything is found again if one of the conditions holds:
 - Total displacement exceeds a certain threshold
 - Big translation, tracker may not keep up with the lip region
 - Within some number of frames from the latest reinitialization
 - Scene may totally change during the tracking

Decoding / Recognition

Use *HTK* for recognition; HTK provides live recognition but only from audio

Needed to modify source code;

- Support multi stream at live recognition
 - Edit live decoding functions
 - Feed with collected live features

Problems with Live Decoding

- Need to send *chunks* of data
- Chunks can be detected by silence detection
- Instead try recognizing short segments of data

Offline Recognition Results

- Audio recognition works well offline
(i.e. Regular HTK process)
- Adding video decreases accuracy, why?
 - Bad visual features
(jumps in frames etc.)
 - Asynchrony in A/V streams?

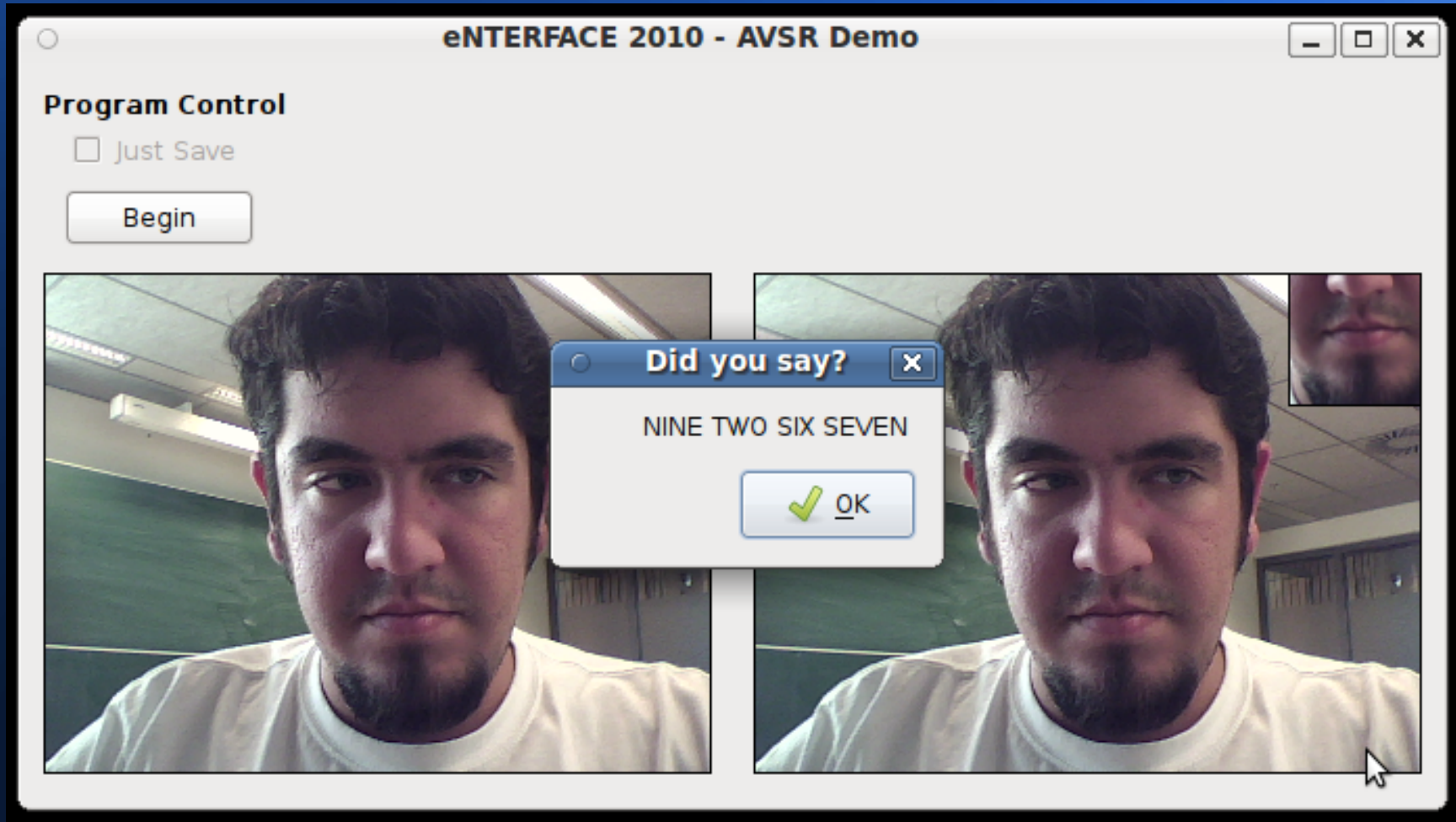
How to Improve?

- Improve ROI tracking
 - Better camera
(only used a regular Laptop Webcam)
- Check A/V asynchrony

Work Done So Far

- Acquire short chunks of data
 - After finishing acquisition,
extract features
(Need more CPU for sync. threads)
- Use extracted features for recognition

Work Done So Far



To be done: Silence Detection

- To make continuous decoding, data should be sent as segments
- Segments can be detected by silence in speech and/or visual inactivity

To Be Done: Tandem Features

- Train classifiers (SVM, NN) for video data
 - Use output of classifiers for tandem features; additional streams

Thanks

IDEAS
&
QUESTIONS ?